Meerkat_Mafia: Multilingual and Cross-Level Semantic Textual Similarity Systems

Abhay Kashyap, Lushan Han, Roberto Yus, Jennifer Sleeman,
Taneeya Satyapanich, Sunil Gandhi and Tim Finin
University of Maryland, Baltimore County
Baltimore, MD 21250 USA
{abhay1,lushan1,ryus,jsleem1,taneeya1,sunilga1,finin}@umbc.edu

Abstract

We describe UMBC's systems developed for the SemEval 2014 tasks on *Multilingual Semantic Textual Similarity (Task 10)* and *Cross-Level Semantic Similarity (Task 3)*. Our best submission in the Multilingual task ranked second in both English and Spanish subtasks using an unsupervised approach. Our best systems for Cross-Level task ranked second in Paragraph-Sentence and first in both Sentence-Phrase and Word-Sense subtask. The system ranked first for the Phrase-Word subtask but was not included in the official results due to a late submission.

1 Introduction

We describe the semantic text similarity systems we developed for two of the SemEval tasks for the 2014 International Workshop on Semantic Evaluation. We developed systems for task 3, Cross-Level Semantic Similarity (Jurgens et al., 2014), and task 10, Multilingual Semantic Textual Similarity (Agirre et al., 2014). A key component in all the systems was an enhanced version of the word similarity system used in our entry (Han et al., 2013b) in the 2013 SemEval Semantic Textual Similarity task.

Our best system in the Multilingual Semantic Textual Similarity task used an unsupervised approach and ranked second in both the English and Spanish subtasks. In the Cross-Level Semantic Similarity task we developed a number of new algorithms and used new linguistic data resources. In this task, our best systems ranked second in the Paragraph-Sentence task, first in the Sentence-Phrase task and first in the Word-Sense task. The system ranked first for the Phrase-Word task but was not included in the official results due to a late submission.

The remainder of the paper proceeds as follows. Section 2 describes our word similarity model and it's wrapper to deal with named entities and out of vocabulary words. Sections 3 and 4 describe how we extended the word similarity model for the specific tasks. Section 5 presents the results we achieved on these tasks along with instances where the system failed. Section 6 highlights our future plans for improving the system.

2 Semantic Word Similarity Model

2.1 LSA Word Similarity Model

Our word similarity model is a revised version of the one we used in the 2013 *SEM semantic text similarity task. This was in turn derived from a system developed for the Graph of Relations project (UMBC, 2013b). For SemEval, we wanted a measure that considered a word's semantics but not its lexical category, e.g., the verb "marry" should be semantically similar to the noun "wife". An online demonstration of a similar model developed for the GOR project is available (UMBC, 2013a), but it lacks some of this version's features.

LSA-based word similarity. LSA Word Similarity relies on the distributional hypothesis that words occurring in the same context tend to have similar meanings (Harris, 1968). LSA relies on the fact that words that are semantically similar (e.g., cat and feline or nurse and doctor) are more likely to occur near one another in text. Thus evidence for word similarity can be computed from a statistical analysis of a large text corpus.

We extracted raw word co-occurrence statistics from a portion of the 2007 crawl of the Web corpus from the Stanford WebBase project (Stanford, 2001). We processed the collection to remove some undesirable elements (text duplica-

Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 416–423,
Dublin, Ireland, August 23-24, 2014.

Word pair	±4 model	±1 model
1 doctor_NN, physician_NN	0.775	0.726
2 car_NN, vehicle_NN	0.748	0.802
3 person_NN, car_NN	0.038	0.024
4 car_NN, country_NN	0.000	0.016
5 person_NN, country_NN	0.031	0.069
6 child_NN, marry_VB	0.098	0.000
7 wife_NN, marry_VB	0.548	0.274
8 author_NN, write_VB	0.364	0.128
9 doctor_NN, hospital_NN	0.473	0.347
10 car_NN, driver_NN	0.497	0.281

Table 1: Examples from the LSA similarity model.

tions, truncated text, non-English text and strange characters) and produced a three billion word corpus of high quality English, which is available online (Han and Finin, 2013).

We performed POS tagging and lemmatization on the corpus using the Stanford POS tagger (Toutanova et al., 2000). Word/term co-occurrences were counted in a moving window of a fixed size that scans the entire corpus. We generated two co-occurrence models using window sizes ± 1 and ± 4 because we observed different natures of the models. ± 1 window produces a context similar to the dependency context used in (Lin, 1998). It provides a more precise context but is only good for comparing words within the same POS. This is because words of different POS are typically surrounded by words in different syntactic forms. In contrast, a context window of ± 4 words allows us to compute semantic similarity between words with different POS.

Examples from our LSA similarity model are given in Table 1. Pairs 1 to 6 illustrate that the measure has a good property of differentiating similar words from non-similar words. Examples 7 and 8 show that the ± 4 model can detect semantically similar words even with different POS while the ± 1 model yields poor results. The pairs in 9 and 10 show that highly related, but not substitutable, words may have a strong similarity and that the ± 1 model is better at detecting them.

Our word co-occurrence models were based on a predefined vocabulary of more than 22,000 common English words and noun phrases. We also added to it more than 2,000 verb phrases extracted from WordNet. The final dimensions of our word co-occurrence matrices are 29,000 $\times$ 29,000 when words are POS tagged. Our vocabulary includes only open-class words, i.e., nouns, verbs, adjectives and adverbs. There are no proper nouns in the vocabulary with the only exception of country names.

Singular Value Decomposition (SVD) has been found to be effective in improving word similarity measures (Landauer and Dumais, 1997). SVD is typically applied to a *word by document* matrix, yielding the familiar LSA technique. In our case, we apply it to our *word by word* matrix (Burgess et al., 1998). Before performing SVD, we transform the raw word co-occurrence count f_{ij} to its log frequency $log(f_{ij}+1)$. We select the 300 largest singular values and reduce the 29K word vectors to 300 dimensions. The LSA similarity between two words is defined as the cosine similarity of their corresponding word vectors after the SVD transformation. See (Han et al., 2013b; Lushan Han, 2014) for examples and more information on the LSA model.

Statistical word similarity measures have limitations. Related words can have similarity scores as high as what similar words get, e.g., "doctor" and "hospital". Word similarity is typically low for synonyms that have many word senses since information about different senses are mashed together (Han et al., 2013a). To address these issues, we augment the similarity between two words using knowledge from WordNet, for example, increasing the score if they are in the same WordNet synset or if one is a direct or two link hypernym of the other. See (Han et al., 2013b) for further details.

2.2 Word Similarity Wrapper

Our word similarity model is restricted to the vocabulary size which only comprises open class words. For words outside of the vocabulary, we can only rely on their lexical features and determine equivalence (which we score as 0 or 1, since a continuous scale makes little sense in this scenario). An analysis of the previous STS datasets show that out-of-vocabulary words account for about $25-45\%$ of the total words. Datasets like MSRpar and headlines lie on the higher end of this spectrum due to the high volume of proper nouns.

In the previous version, we computed a character bigram overlap score given by

$$characterBigramScore = \frac{|A \cap B|}{|A \cup B|}$$

where A and B are the set of bigrams from the first and second word respectively. We compare this

against a preset threshold (0.8) to determine equivalence. While this is reasonable for named entities, it is not the best approach for other classes.

Named Entities. The wrapper is extended to handle all classes of named entities that are included in Stanford CoreNLP (Finkel et al., 2005). We use heuristic rules to compute the similarity between two numbers or two dates. To handle named entity mentions of people, locations and organizations, we supplement our character bigram overlap method with the DBpedia Lookup service (Mendes et al., 2011). For each entity mention, we select the DBpedia entity with the most inlinks, which serves as a good estimate of popularity or significance (Syed et al., 2010). If the two named entity mentions map to identical DBpedia entities, we lower our character bigram overlap threshold to 0.6.

OOV words. As mentioned earlier, when dealing with out-of-vocabulary words, we only have its lexical features. A straightforward approach is to simply get more context for the word. Since our vocabulary is limited, we need to use external dictionaries to find the word. For our system, we use Wordnik (Davidson, 2013), which is a compilation of several dictionaries including The American Heritage Dictionary, Wikitionary and WordNet. Wordnik provides a REST API to access several attributes for a given word such as it's definitions, examples, related words etc. For out of vocabulary words, we simply retrieve the word pair's top definitions and supply it to our existing STS system (UMBC, 2013a) to compute its similarity. As a fallback, in case the word is absent even in Wordnik, we resort to our character bigram overlap measure.

3 Multilingual Semantic Text Similarity

3.1 English STS

For the 2014 STS-English subtask we submitted three runs. They all used a simple term alignment strategy to compute sentence similarities. The first run was an unsupervised approach that used the basic word-similarity model for term alignment. The next two used a supervised approach to combine the scores from the first run with alignment scores using the enhanced word-similarity wrapper. The two runs differed in their training.

Align and Penalize Approach. The *pairingWord* run was produced by the same Align-and-Penalize system (Han et al., 2013b) that we used in the 2013 STS task with only minor changes. The biggest change is that we included a small list of disjoint concepts (Han et al., 2013b) that are used in the penalization phase, such as {*piano*, *violin*} and {*dog*, *cat*}. The disjoint concepts were manually collected from the MSRvid dataset provided by the 2012 STS task because we still lack a reliable general method to automatically produce them. The list only contains 23 pairs, which can be downloaded at (dis, 2014).

We also slightly adjusted our stopword list. We removed a few words that appear in the trial datasets of 2013 STS task (e.g., *frame*) but we did not add any new stopwords for this year's task. All the changes are small and we made them only in the hope that they can slightly improve our system.

Unlike machine learning methods that require manually selecting an appropriate trained model for a particular test dataset, our unsupervised Align-and-Penalize system is applied uniformly to all six test datasets in 2014 STS task, namely, deft-forum, deft-news, headlines, images, OnWN and tweet-news. It achieves the second best rank among all submitted runs.

Supervised Machine Learning. Our second and third runs used machine learning approaches similar to those we developed for the 2013 STS task but with significant changes in both pre-processing and the features extracted.

The most significant pre-processing change was the use of Stanford coreNLP (Finkel et al., 2005) tool for tokenization, part-of-speech tagging and identifying named entity mentions. For the *tweet-news* dataset we also removed the hashtag symbol ('#') prior to applying the Stanford tools. We use only open class words and named entity mentions and remove all other tokens.

We align tokens between two sentences based on the updated word similarity wrapper that was described in Section 2.2. We use information content from Google word frequencies for word weights similar to our approach last year. The alignment process is a many-to-one mapping similar to the Align and Penalize approach and two tokens are only aligned if their similarity is greater than 0.1. The sentence similarity score is then computed as the average of the scores of their aligned tokens. This score, along with the Align and Penalize approach score, are used as features to train support vector regression (SVR) models.

We use an epsilon SVR with a radial basis kernel function and use a grid search to get the optimal parameter values for cost, gamma and epsilon. We use datasets from the previous STS tasks as training data and the two submitted runs differ in the choice of their training data.

The first approach, named *Hulk*, is an attempt to use a generic model trained on a large data set. The SVR model uses a total of 3750 sentence pairs (1500 from MSRvid, 1500 from MSRpar and 750 from headlines) for training. Datasets like SMT were excluded due to poor quality.

The second approach, named *Super Saiyan*, is an attempt at domain specific training. For OnWN, we used 1361 sentence pairs from previous OnWN dataset. For Images, we used 1500 sentence pairs from MSRvid dataset. The others lacked any domain specific training data so we used a generic training dataset comprising 5111 sentence pairs from MSRvid, MSRpar, headlines and OnWN datasets.

3.2 Spanish STS

As a base-line for this task we first considered translating the Spanish sentences to English and running the same systems explained for the English Subtask (i.e., *pairingWord* and *Hulk*). The results obtained applying this approach to the provided training data gave a correlation of 0.777 so, we selected this approach (with some improvements) for the competition.

Translating the sentences. For the automatic translation of the sentences from Spanish to English we used the Google Translate API[1], a free, multilingual machine-translation product by Google. Google Translate presents very accurate translations for European languages by using statistical machine translation (Brown et al., 1990) where the translations are generated on the basis of statistical models derived from bilingual text corpora. In fact, Google used as part of this corpora 200 billion words from United Nations documents that are typically published in all six official UN languages, including English and Spanish.

In the experiments performed with the trial data we manually evaluated the quality of the translations (one of the authors is a native Spanish speaker). The overall translation was very accurate but some statistical anomalies, incorrect translations due to the abundance of a specific sense of a word in the training set, appeared.

I1: Las costas o costa de un mar, lago o extenso río es la tierra a lo largo del borde de estos.

T11: Costs or the cost of a sea, lake or wide river is the land along the edge of these.

T12: Coasts or the coast of a sea, lake or wide river is the land along the edge of these.

T13: Coasts or the coast of a sea, lake or wide river is the land along the border of these.

...

Figure 1: Three of the English translations for the Spanish sentence I1.

On one hand, some homonym words are wrongly translated. For example, the Spanish sentence *"Las **costas o costa** de un mar [...]"* was translated to *"**Costs or the cost** of a sea [...]"*. The Spanish word *costa* has two different senses: "coast" (the shore of a sea or ocean) and "'cost" (the property of having material worth). On the other hand, some words are translated preserving their semantics but with a slightly different meaning. For example, the Spanish sentence *"Un **cojín** es una funda de tela [...]"* was correctly translated to *"A **cushion** is a fabric cover [...]"*. However, the Spanish sentence *"Una almohada es un **cojín** en forma rectangular [...]"* was translated to *"A pillow is a rectangular **pad** [...]"*[2].

Dealing with statistical anomalies. The aforementioned problem of statistical machine translation caused a slightly adverse effect when computing the similarity of two English (translated from Spanish) sentences with the systems explained in Section 3.1. Therefore, we improved the direct translation approach by taking into account the different possible translations for each word in a Spanish sentence. For that, our system used the information provided by the Google Translate API, that is, all the possible translations for every word of the sentence along with a popularity value. For each Spanish sentence the system generates all its possible translations by combining the different possible translations of each word. For example, Figure 1 shows three of the English sentences generated for a given Spanish sentence from the trial data.

As a way of controlling the combinatorial explosion of this step, especially for long sentences, we limited the maximum number of generated

[1] http://translate.google.com

[2] Notice that both Spanish sentences used the term *cojín* that should be translated as *cushion* (the Spanish word for *pad* is *almohadilla*).

sentences for each Spanish sentence to 20 and we only selected words with a popularity greater than 65. We arrived at the popularity threshold through experimentation on every sentence in the trial data set. After this filtering, our input for the "news" and "wikipedia" tests went from 480 and 324 pairs of sentences to 5756 and 1776 pairs, respectively.

Given a pair of Spanish sentences, $I1$ and $I2$, and the set of possible translations generated by our system for each sentence, $T_{I1} = \{T_{11}, T_{12}, T_{13}, \ldots, T_{1n}\}$ and $T_{I2} = \{T_{21}, T_{22}, \ldots, T_{2m}\}$, we compute the similarity between them by using the following formula:

$$SimSPA(I1, I2) = \frac{\sum_{i=1}^{n}\sum_{j=1}^{m} SimENG(T_{1i}, T_{2j})}{n * m}$$

where $SimENG(x, y)$ computes the similarity of two English sentences using our existing STS system (Han et al., 2013b).

For the final competition we submitted three runs. The first (*Pairing* in Table 3) used the *pairingWord* system with the direct translation of the Spanish sentences to English. The second run (*PairingAvg* in Table 3) used the formula for $SimSPA(x, y)$ based on $SimENG(x, y)$ with the *pairingWord* system. Finally, the third one (*Hulk* in Table 3) used the *Hulk* system with the direct translation.

4 Cross Level Similarity

4.1 Sentence to Paragraph/Phrase

We used the three systems developed for the English sentence similarity subtask and described in Section 3.1 for both the sentence to paragraph and sentence to phrase subtasks, producing three runs. The model for *Hulk* remained the same (trained on 3750 sentence pairs from MSRvid, MSRpar and headlines dataset) but the *SuperSaiyan* system, which is the domain specific approach, used the given train and trial text pairs (about 530) for the respective subtasks as training to generate task specific models.

4.2 Phrase to Word

In our initial experiments, we directly computed the phrase-word pair similarity using our English STS. This yielded a very low correlation of 0.239 for the training set, primarily due to the absence of these phrases and words in our vocabulary. To address this issue, we used external sources to obtain more contextual information and extracted several features.

Dictionary features. We used Wordnik as a dictionary resource and retrieved definitions and usage examples for the word. We then used our English STS system to measure the similarity between these and the given phrase to extract two features.

Web search features. These features were based on the hypothesis that if a word and phrase have similar meanings, then a web search that combines the word and phrase should return similar documents when compared to a web search for each individually.

We implemented this idea by comparing results of three search queries: the word alone, the phrase alone, and the word and phrase together.

Using the Bing Search API (BIN, 2014), we retrieved the top five results for each search, indexed them with Lucene (Hatcher et al., 2004), and extracted term frequency vectors for each of the three search result document sets. For the phrase *'spill the beans'* and word *'confess'*, for example, we built a Lucene index for the set of documents retrieved by a Bing search for 'spill the beans', 'confess', and 'spill the beans confess'. We calculated the similarity of pairs of search result sets using the cosine similarity (1) of their term frequency vectors.

$$CosineSimilarity = \frac{\sum_{i=1}^{n} V1_i \times V2_i}{\sqrt{\sum_{i=1}^{n}(V1_i)^2} \times \sqrt{\sum_{i=1}^{n}(V2_i)^2}} \quad (1)$$

We calculated the mean and minimum similarity of pairs of results for the phrase and phrase+word searches. These features were extracted from the provided training set and used in conjunction with the dictionary features to train an SVM regression model to predict similarity scores.

We observed this method can be problematic when a word or phrase has multiple meanings. For example, *'spill the beans'* relates to *'confessing'* but it is also the name of a coffee shop and a soup shop. A mix of these pages do get returned by Bing and reduces the accuracy of our results. However, we found that this technique often strengthens evidence of similarity enough that it improves our overall accuracy when used in combination with our dictionary features.

Dante#n#1: an Italian poet famous for writing the Divine Comedy that describes a journey through Hell and purgatory and paradise guided by Virgil and his idealized Beatrice

writer#n#1: writes books or stories or articles or the like professionally for pay

generator#n#3: someone who originates or causes or initiates something, "he was the generator of several complaints"

author#v#1: be the author of, "She authored this play"

Figure 2: The WordNet sense for *Dante#n#1* and the three *author#n* senses.

4.3 Word to Sense

For this subtask, we used external resources to retrieve more contextual information. For a given word, we retrieved its synonym set from WordNet along with their corresponding definitions. We retrieved the WordNet definition for the word sense as well. For example, given a word-sense pair (*author#n*, *Dante#n#1*), we retrieved the synset of *author#n* (*writer.n.01*, *generator.n.03*, *author.v.01*) along with their WordNet definitions and the sense definition of *Dante#n#1*. Figure 2 shows the WordNet data for this example.

By pairing every combination of the word's synset and their corresponding definitions with the sense's surface form and definition, we created four features. For each feature, we used our English STS system to compare their semantic similarity and kept the maximum score as feature's value.

We found that about 10% of the training dataset's words fell outside of WordNet's vocabulary. Examples of missing words included many informal or "slang" words like *kegger*, *crackberry* and *post-season*. To address this, we used Wordnik to retrieve the word's top definition and computed its similarity with the sense. This reduced the out-of-vocabulary words to about 2% for the training data. Wordnik thus gave us two additional features: the maximum semantic similarity score of word-sense using Wordnik's additional definitions for all words and for just the out-of-vocabulary words. We used these features to train an SVM regression model with the provided training set to predict similarity scores.

Dataset	Pairing	Hulk	SuperSaiyan
deft-forum	0.4711 (9)	0.4495 (15)	0.4918 (4)
deft-news	0.7628 (8)	**0.7850 (1)**	0.7712 (3)
headlines	0.7597 (8)	0.7571 (9)	0.7666 (2)
images	0.8013 (7)	0.7896 (10)	0.7676 (18)
OnWN	**0.8745 (1)**	0.7872 (18)	0.8022 (12)
tweet-news	0.7793 (2)	0.7571 (7)	0.7651 (4)
Weighted Mean	**0.7605 (2)**	**0.7349 (6)**	**0.7410 (5)**

Table 2: Performance of our three systems on the six English test sets.

Dataset	Pairing	PairingAvg	Hulk
Wikipedia	0.6682 (12)	0.7431 (6)	0.7382 (8)
News	0.7852 (12)	**0.8454 (1)**	0.8225 (6)
Weighted Mean	**0.7380 (13)**	**0.8042 (2)**	**0.7885 (5)**

Table 3: Performance of our three systems on the two Spanish test sets.

5 Results

Multilingual Semantic Text Similarity. Table 2 shows the system performance for the English STS task. Our best performing system ranked second [3], behind first place by only 0.0005. It employs an unsupervised approach with no training data required. The supervised systems that handled named entity recognition and out-of-vocabulary words performed slightly better on datasets in the news domain but still suffered from noise due to diverse training datasets.

Table 3 shows the performance for the Spanish subtask. The best run achieved a weighted correlation of 0.804, behind first place by only 0.003. The *Hulk* system was similar to the *Pairing* run and used only one translation per sentence. The performance boost could be attributed to large number of named entities in the News and Wikipedia datasets.

Cross Level Similarity. Table 4 shows our performance in the Cross Level Similarity tasks. The Paragraph-Sentence and Sentence-Phrase yielded good results (ranked second and first respectively) with our English STS system because of sufficient amount of textual information. The correlation scores dropped as the granularity level of the text got finer.

The Phrase-Word run achieved a correlation of 0.457, the highest for the subtask. However, an incorrect file was submitted prior to the deadline

[3] An incorrect file for 'deft-forum' dataset was submitted. The correct version had a correlation of 0.4896 instead of 0.4710. This would have placed it at rank 1 overall.

				Wordnik		BingSim			Score		
ID	S1	S2	Baseline	Definitions	Example	Sim	Avg	Min	SVM	GS	Error
Idiomatic-212	spill the beans	confess	0	0	0	0.0282	0.1516	0.1266	0.5998	4.0	3.4002
Idiomatic-292	screw the pooch	mess up	0	0.04553	0.0176	0.0873	0.4238	0.0687	0.7185	4.0	3.2815
Idiomatic-273	on a shoogly peg	insecure	0	0.0793	0	0.0846	0.3115	0.1412	0.8830	4.0	3.1170
Slang-115	wacky tabaccy	cannabis	0	0	0	0.0639	0.4960	0.1201	0.5490	4.0	3.4510
Slang-26	pray to the porcelain god	vomiting	0	0	0	0.0934	0.5275	0.0999	0.6452	4.0	3.3548
Slang-79	rock and roll	commence	0	0.2068	0.0720	0.0467	0.5106	0.0560	0.8820	4.0	3.1180
Newswire-160	exercising rights under canon law	lawyer	0.0044	0.6864	0.0046	0.3642	0.4990	0.2402	3.5562	0.5	3.0562

Table 5: Examples where our algorithm performed poorly and the scores for individual features.

Dataset	Pairing	Hulk	SuperSaiyan	WordExpand
Para.-Sent.	0.794 (10)	0.826 (4)	0.834 (2)	
Sent.-Phrase	0.704 (14)	0.705 (13)	**0.777 (1)**	
Phrase-Word				**0.457 (1)**
Word-Sense				**0.389 (1)**

Table 4: Performance of our systems on the four Cross-Level Subtasks.

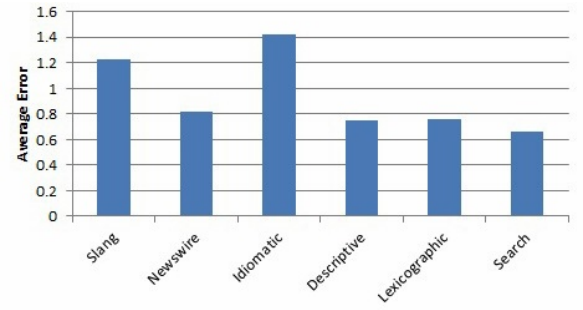

Figure 3: Average error with respect to category.

ID	word	sense key	sense number	predicted	gold
80	cheese#n	moolah%1:21:00::	moolah#n#1	0.78	4
377	bone#n	chalk%1:07:00::	chalk#n#2	1.52	4
441	wasteoid#n	drug user%1:18:00::	drug user#n#1	0.78	3

Table 6: Examples where our system performed poorly.

which meant that this was not included in the official results. Figure 3 shows the average error (measured as the average deviation from the gold standard) across different categories for phrase to word subtask. Our performance is slightly worse for slang and idiomatic categories when compared to others which is due to two reasons: (i) the semantics of idioms is not compositional, reducing the effectiveness of a distributional similarity measure and (ii) dictionary-based features often failed to find definitions and/or examples of idioms. Table 5 shows some of the words where our algorithm performed poorly and their scores for individual features.

The Word-Sense run ranked first in the subtask with a correlation score of 0.389. Table 6 shows some of the word-sense pairs where the system performed poorly. Our system only used Wordnik's top definition which was not always the right one to use to detect the similarity. For example, the first definition of *cheese#n* is "a solid food prepared from the pressed curd of milk" but there is a latter, less prominent one, which is "money". A second problem is that some words, like *wasteoid#n*, were absent even in Wordnik.

Using additional online lexical resources to include more slangs and idioms, like the *Urban Dictionary* (Urb, 2014), could address these issues. However, care must be taken since the quality of some content is questionable. For example, the Urban Dictionary's first definition of "programmer" is "An organism capable of converting caffeine into code".

6 Conclusion

We described our submissions to the *Multilingual Semantic Textual Similarity (Task 10)* and *Cross-Level Semantic Similarity (Task 3)* tasks for the 2014 International Workshop on Semantic Evaluation. Our best runs ranked second in both English and Spanish subtasks for Task 10 while ranking first in Sentence-Phrase, Phrase-Word, Word-Sense tasks and second in Paragraph-Sentence subtasks for Task 3. Our success is attributed to a powerful word similarity model based on LSA word similarity and WordNet knowledge. We used new linguistic resources like Wordnik to improve our existing system for the Phrase-Word and Word-Sense tasks and plan to include other resources like "Urban dictionary" in the future.

Acknowledgements

This research was supported by awards 1228198, 1250627 and 0910838 from the U.S. National Science Foundation.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. SemEval-2014 Task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, Dublin, Ireland.

2014. BING search API. http://bing.com/developers-/s/APIBasics.html.

Peter F Brown, John Cocke, Stephen A Della Pietra, Vincent J Della Pietra, Fredrick Jelinek, John D Lafferty, Robert L Mercer, and Paul S Roossin. 1990. A statistical approach to machine translation. *Computational linguistics*, 16(2):79–85.

Curt Burgess, Kay Livesay, and Kevin Lund. 1998. Explorations in context space: Words, sentences, discourse. *Discourse Processes*, 25(2-3):211–257.

Sara Davidson. 2013. Wordnik. *The Charleston Advisor*, 15(2):54–58.

2014. Disjoint concept pairs. http://semanticweb-archive.cs.umbc.edu/disjointConcepts.txt.

Jenny Rose Finkel, Trond Grenager, and Christopher D. Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *43rd Annual Meeting of the ACL*, pages 363–370.

Lushan Han and Tim Finin. 2013. UMBC webbase corpus. http://ebiq.org/r/351.

Lushan Han, Tim Finin, Paul McNamee, Anupam Joshi, and Yelena Yesha. 2013a. Improving Word Similarity by Augmenting PMI with Estimates of Word Polysemy. *IEEE Trans. on Knowledge and Data Engineering*, 25(6):1307–1322.

Lushan Han, Abhay L. Kashyap, Tim Finin, James Mayfield, and Johnathan Weese. 2013b. UMBC_EBIQUITY-CORE: Semantic Textual Similarity Systems. In *2nd Joint Conf. on Lexical and Computational Semantics*. ACL, June.

Zellig Harris. 1968. *Mathematical Structures of Language*. Wiley, New York, USA.

Erik Hatcher, Otis Gospodnetic, and Michael McCandless. 2004. *Lucene in action*. Manning Publications Greenwich, CT.

David Jurgens, Mohammad Taher Pilehvar, and Roberto Navigli. 2014. SemEval-2014 Task 3: Cross-Level Semantic Similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval-2014)*, Dublin, Ireland.

Thomas K Landauer and Susan T Dumais. 1997. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proc. 17th Int. Conf. on Computational Linguistics*, pages 768–774, Montreal, CN.

Lushan Han. 2014. *Schema Free Querying of Semantic Data*. Ph.D. thesis, University of Maryland, Baltimore County.

Pablo N Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. 2011. Dbpedia spotlight: shedding light on the web of documents. In *7th Int. Conf. on Semantic Systems*, pages 1–8. ACM.

Stanford. 2001. Stanford WebBase project. http://bit.ly/WebBase.

Zareen Syed, Tim Finin, Varish Mulwad, and Anupam Joshi. 2010. Exploiting a Web of Semantic Data for Interpreting Tables. In *Proceedings of the Second Web Science Conference*, April.

Kristina Toutanova, Dan Klein, Christopher Manning, William Morgan, Anna Rafferty, and Michel Galley. 2000. Stanford log-linear part-of-speech tagger. http://nlp.stanford.edu/software/tagger.shtml.

UMBC. 2013a. Semantic similarity demonstration. http://swoogle.umbc.edu/SimService/.

UMBC. 2013b. Umbc graph of relations project. http://ebiq.org/j/95.

2014. Urban dictionary. http://urbandictionary.com/.